Amendments to the Specification:

Please replace paragraph 9 of the specification with the following rewritten paragraph:

5. a single FSOURCE ~~fatwire~~ IO that supplies current from a chip signal IO to all the fuse macros during fuse blow.

Please replace paragraph 10 of the specification with the following rewritten paragraph:

The wire associated with the FSOURCE is referred to as a "fatwire" (depicted as a relatively heavier connections in the Figures) because the type of wire used in an FSOURCE connection which has a width that can be over 100 times that of a minimum width wire in the circuit in order to carry sufficient current while limiting voltage drop per unit length during programming of fuses.

Please delete paragraphs 11 and 15-19 of the specification.

Please insert the following six paragraphs between paragraphs 37 and 38 of the specification (i.e. after paragraph 37 and before paragraph 38):

The typical arrangement of a fuse domain is depicted schematically in Figure 1 which shows a single fuse controller connected to primary, secondary and tertiary fuse macros which may contain up to 63, 3 and 1 (e.g. 0-63, 0-3 and 0-1, respectively) additional macros each, giving a total of 64, 4 and 2 macros each, respectively. As can be seen in the figure, the primary, secondary and tertiary fuse macros are connected to a single FSOURCE IO cell, which is in turn connected to an FSOURCE C4 pad. In general, an electronic fuse domain will contain a set of primary fuses, an FSOURCE power connection, and other overhead.

Figure 2 illustrates typical placement of a fuse domain that clusters around an FSOURCE IO cell. As can be seen, the cluster of fuses is so dense that IO cells cannot be placed near the C4 pads on top of this cluster and must be located at a distance. Because the distance from the C4

pads to any IO cell is so large in this arrangement, the resistance through a wire connecting a C4 pad to an IO cell is too great to make the C4 pad usable.

Figure 3 further and more generally illustrates this problem. In Figure 3, a large placement obstruction (for example, a fixed block or a group of fixed blocks such as a group of IO cells) is located at a position where it would be preferable to locate primary fuse macros. Since the fuse macro once programmed has no effect on the functionality of the chip, in theory its placement away from the other fuses and on the other side of the placement obstruction, as shown, is allowable. However, placement of the primary fuse macro at this location makes the distance from the fuse macro to the FSOURCE IO cell too great to meet voltage drop requirements (i.e. the distance is too large or "excessively distal" to meet the V = IR drop adequate for reliable fuse programming, or, more generally, to reliably perform any other functions, for example, due to noise, signal propagation, time, etc.). As a result the fuses inside an excessively distal fuse macro may fail to be blown during testing.

One potential solution to the problem illustrated in Figure 3 is to split the fuse domain into multiple domains as shown in Figure 4. For example, if a single domain is split into two domains, instead of dealing with one large cluster of fuse macros that must be placed very close together around one FSOURCE IO cell, two smaller clusters of fuse macros may be placed around two FSOURCE IO cells. These smaller clusters can be placed independently and may be spaced further apart within a cluster. Figure 4 illustrates this possibility, showing two separate fuse domains located on opposite sides of a placement obstruction. While this solution alleviates some of the floorplanning problems caused by utilizing a single large cluster, this type of design also has some drawbacks. For example:

1. In addition to the primary fuse macros, each domain requires one fuse controller, one FSOURCE fatwire IO, 4 secondary fuse macros, and 2 tertiary fuse macros. Therefore, splitting a full domain of 64 primary fuse macros into two domains of 32 primary fuse macros (or other allocations) adds all the overhead of a fuse domain, i.e. a second fuse controller, a second FSOURCE fatwire IO cell, 4 additional secondary fuse macros, and 2 additional tertiary fuse macros. (In Figure 4, this overhead is simplified and depicted as a fuse controller, an FSOURCE fatwire IO cell, a single primary fuse macro, a single secondary fuse macro, and a single tertiary

fuse macro.)

2. Fuse domains may not share fuse macros. If one domain uses all of its fuses and the other domain is using none, the domain that is fully utilized cannot use any of the macros from the empty domain if it should require more fuses. This scenario could lead to discarding product that might otherwise have been repaired, since defects on a chip tend to cluster in regions.

Please replace paragraph 21, with the following rewritten paragraph:

The present invention provides a solution for the floorplanning problems associated with placement of fuses on a semiconductor chip. According to the practice of the invention, instead of breaking up an entire fuse domain into multiple domains in order to provide flexibility of floorplanning, the FSOURCE connection associated with the fuse domain is split into multiple nets. In other words, in the practice of the present invention, a fuse domain may contain a plurality of FSOURCE ~~fatwire~~ IO cells, and a primary fuse macro within a domain may be connected to any of the FSOURCE ~~fatwire~~ IO cells in the domain. Therefore, a fuse macro (primary, secondary, or tertiary) may be located at any distance from the fuse controller, and provision must be made only for the wire that connect the distal fuse macro to the fuse controller (plus the additional FSOURCE IO cell and C4 pad, both of which are co-located with the distal primary fuse macro). Fuse macros in a fuse domain thus are relatively mobile and may be flexibly located on a microchip without increasing the number of fuse controllers and attendant overhead. As illustrated in Figure 5, only the wire connecting the fuse controller to the distal primary fuse macro traverses the area occupied by a placement obstruction, and the distal primary fuse macro and its supplemental FSOURCE IO cell and C4 pad are placed well outside the placement obstruction.

Please replace paragraph 38, with the following rewritten paragraph:

Figure 5 is a schematic depiction of one embodiment of the present invention, in which the minimal ~~overhead~~ solution required to solve a fuse placement problem is illustrated. As can

be seen in Figure 5, one primary fuse macro is located at a distance from the other fuse macros in the domain due to a placement obstruction. A similar scenario was depicted in Figure 3; however, in Figure 3, the distance between the distal fuse macro and the FSOURCE IO ~~fatwire~~ cell was too great to allow the distal fuse to be reliably utilized. Problems associated with routing a fatwire are crowding on the chip surface and the inability to blow a fuse located at a distance. In contrast, in the practice of the present invention, the distal primary fuse macro may be utilized because a second, supplemental FSOURCE IO ~~fatwire~~ cell is incorporated into the fuse domain design. The supplemental FSOURCE IO ~~fatwire~~ cell is proximate to (e.g. within a distance that suitably limits the v = IR voltage drop so that it is not "excessively distal") and operatively connected to the distal primary fuse macro. As used herein, "operatively connected" means that the FSOURCE IO ~~fatwire~~ cell is connected to the distal primary fuse macro in a manner that allows the FSOURCE IO ~~fatwire~~ cell to carry out its normal function, i.e. to provide current to blow the primary fuse macro. As can be seen, the only additional overhead required to permit movement of the primary fuse macro to the distal site is the second FSOURCE IO ~~fatwire~~ cell and the associated C4 pad. Thus, by including additional FSOURCE IO ~~fatwire~~ cells in the design of a fuse domain, it is possible to place primary fuse macros at convenient locations on the semiconductor chip. The fuse controller can control fuse programming over a normal (e.g. not a fatwire) connector since control currents are small.

Please replace paragraph 39, with the following rewritten paragraph:

In a preferred embodiment, the present invention will be implemented in fuse insertion software. The software is modified to break up the fuses in the fuse domain into a user-specified number of subgroups. Each subgroup of fuses is connected to its own FSOURCE ~~fatwire~~ IO <u>cell</u>. A threshold will be used to determine the maximum number of fuse macros per subgroup during fuse insertion. This threshold will be set to a default for each new design and can be changed based on learning during floorplanning for the next iteration of the design.